

Patent Application No. 09/338,035

IN THE CLAIMS:

Please amend claims 1, 8 and 9 as follows:

1 1. (currently amended) A method for processing tasks in a data
2 processing system including a microprocessor and an instruction cache
3 wherein tasks of ~~different~~ various types are defined in the system, each
4 task type having code associated therewith, the tasks being processed out
5 of sequential order by loading the associated code into the instruction
6 cache for execution on the microprocessor, the method comprising the step
7 of

8 placing the tasks of ~~the~~ same task type into a batch such that the
9 tasks in a batch are processed before processing the next ordered task;
10 and

11 wherein the tasks of the same task type use same code in the
12 instruction cache.

1 2. (original) A method as claimed in claim 1 wherein the code
2 associated with at least one type of task fits within the instruction
3 cache, the method comprises the further steps of: processing such a
4 task by loading the associated code into the instruction cache and
5 executing the code on the microprocessor, and, on a determination
6 that there is a further task of like type in the batch, executing the
7 loaded code to process the further task.

1 3. (original) A method as claimed in claim 1 wherein the code
2 associated with at least one type of task is not capable of being
3 loaded as a whole into the instruction cache, the code being
4 logically divided at one or more break points into two or more
5 portions, and wherein during processing of such a task, the method
6 comprises the further steps of responding to a break point defined
7 within a first portion of the code to schedule a further task for
8 future execution of a second portion of the code.

1 4. (original) A method as claimed in claim 3 wherein the
2 further scheduled task is placed in a batch of like tasks.

1 5. (original) A method as claimed in claim 3 wherein each of
2 portions of code defines an atomic operation.

Patent Application No. 09/338,035

1 6. (previously presented) A method as claimed in claim 1,
2 wherein a task is placed in a batch at the time the task is
3 scheduled.

1 7. (previously presented) A method as claimed in claim 1
2 wherein the tasks are managed as a queue.

1 8. (currently amended) A computer program product comprising a
2 computer usable medium having computer readable program code means
3 embodied in the medium for processing tasks in a data processing
4 system, the data processing system including a microprocessor and an
5 instruction cache and wherein tasks of ~~different~~ various types are
6 defined in the system, each task type having code associated
7 therewith, the tasks being processed out of sequential order by
8 executing the associated code on the microprocessor, the program code
9 means comprising code means for scheduling tasks of ~~the~~ same type
10 into a batch such that tasks in a batch are processed before
11 processing the next ordered task, wherein the tasks of the same type
12 use the same program code.

1 9. (currently amended) Data processing apparatus comprising a
2 microprocessor and an instruction cache wherein tasks of ~~different~~ various
3 types are defined in the system, each task type having code associated
4 therewith, the apparatus including:
5 means for processing the tasks out of sequential order by loading
6 the associated code into the instruction cache for execution on the
7 microprocessor; and
8 means for scheduling tasks of ~~the~~ same type into a batch, wherein
9 the means for processing the tasks is operable to process the tasks in a
10 batch before processing the next ordered task; and
11 wherein the tasks of the same type use the same program code.

1 10. (original) Data processing apparatus as claimed in claim 9
2 wherein the microprocessor and i-cache are embodied on a single chip.

1 11. (previously presented) A method for scheduling tasks in a task
2 queue, the method comprising:
3 identifying a new task to be scheduled in the task queue;

Patent Application No. 09/338,035

4 determining if the task queue includes a cached task that requires
5 the same code to process the cached task and the new task; and
6 batching the new task with the cached task if the task queue
7 includes the cached task that requires the same code to process the cached
8 task and the new task.

1 12. (previously presented) The method of claim 11, further
2 comprising adding the new task to the end of the queue if the task queue
3 does not include the cached task that requires the same code to process
4 the cached task as the new task.

1 13. (previously presented) The method of claim 11, further
2 comprising:
3 loading task code for processing the cached task into an instruction
4 cache;
5 executing the task code for processing the cached task in the
6 instruction cache; and
7 executing the task code for processing the new task in the
8 instruction cache without loading new code into the instruction cache.

1 14. (previously presented) The method of claim 13, further
2 comprising:
3 determining if the task code is capable of fully loading into the
4 instruction cache; and
5 if the task code is not capable of fully loading into the
6 instruction cache, logically dividing the task code such that at least one
7 atomic portion of the task code will fully load in the instruction cache.

1 15. (previously presented) A computer program product embodied in a
2 tangible media comprising:
3 computer readable program codes coupled to the tangible media for
4 scheduling tasks in a task queue, the computer readable program codes
5 configured to cause the program to:
6 identify a new task to be scheduled in the task queue;
7 determine if the task queue includes a cached task that requires the
8 same code to process the cached task and the new task; and
9 batch the new task with the cached task if the task queue includes

Patent Application No. 09/338,035

10 the cached task that requires the same code to process the cached task and
11 the new task.

1 16. (previously presented) The computer program product of claim
2 15, wherein the program code is further configured to cause the program to
3 add the new task to the end of the queue if the task queue does not
4 include the cached task that requires the same code to process the cached
5 task as the new task.

1 17. (previously presented) The computer program product of claim
2 15, wherein the program code is further configured to cause the program
3 to:
4 load task code for processing the cached task into an instruction
5 cache;
6 execute the task code for processing the cached task in the
7 instruction cache; and
8 execute the task code for processing the new task in the instruction
9 cache without loading new code into the instruction cache.

1 18. (previously presented) The computer program product of claim
2 17, wherein the program code is further configured to cause the program
3 to:
4 determine if the task code is capable of fully loading into the
5 instruction cache;
6 if the task code is not capable of fully loading into the
7 instruction cache, logically divide the task code such that at least one
8 atomic portion of the task code will fully load in the instruction cache.